UNIVERSIDAD NACIONAL DE INGENIERIA

FACULTAD DE ELECTROTECNIA Y COMPUTACION

PROPUESTA DE APLICACION WEB DEL EXPEDIENTE ACADÉMICO PARA LOS DOCENTES DE LA FACULTAD DE ELECTROTECNIA Y COMPUTACION DE LA UNIVERSIDAD NACIONAL DE INGENIERIA

Manuales de Usuario y del Programador

Br. Sobeyda Azucena García Aguirre	2007-21488
Br. Verónica Auxiliadora Aguilar Largaespada	2007-22302

Como requisito para optar al Título de:

Ingeniero en Computación

Tutor:

MSc. Ing. Gloria Talía Flores Quintana

Nicaragua, Agosto, 2019

MANUAL DE USUARIO

INTRODUCCIÓN

El documento que se muestra a continuación muestra los manuales de usuario y programador, de la aplicación web del expediente académico para los docentes de la Facultad de Electrotecnia y Computación de la Universidad Nacional de Ingeniería

Un **manual de usuario** es un conjunto de técnicas de uso y recomendaciones que acompañan a un producto en el momento de la adquisición para que el **usuario** pueda hacer un uso adecuado y eficiente del mismo. por lo general, este documento está redactado por un escritor técnico, como por ejemplo los programadores del sistema o los directores

El estudio, así como la aplicación desarrollada integra un manual de usuario, el cual servirá de guía para hacer uso del sistema.

Para empezar a explicar cómo utilizar la aplicación desarrollada lo primero es mostrar la pantalla principal que contiene las credenciales (nombre y clave) para acceder a la misma.

DESARROLLO DEL MANUAL DE USUARIO

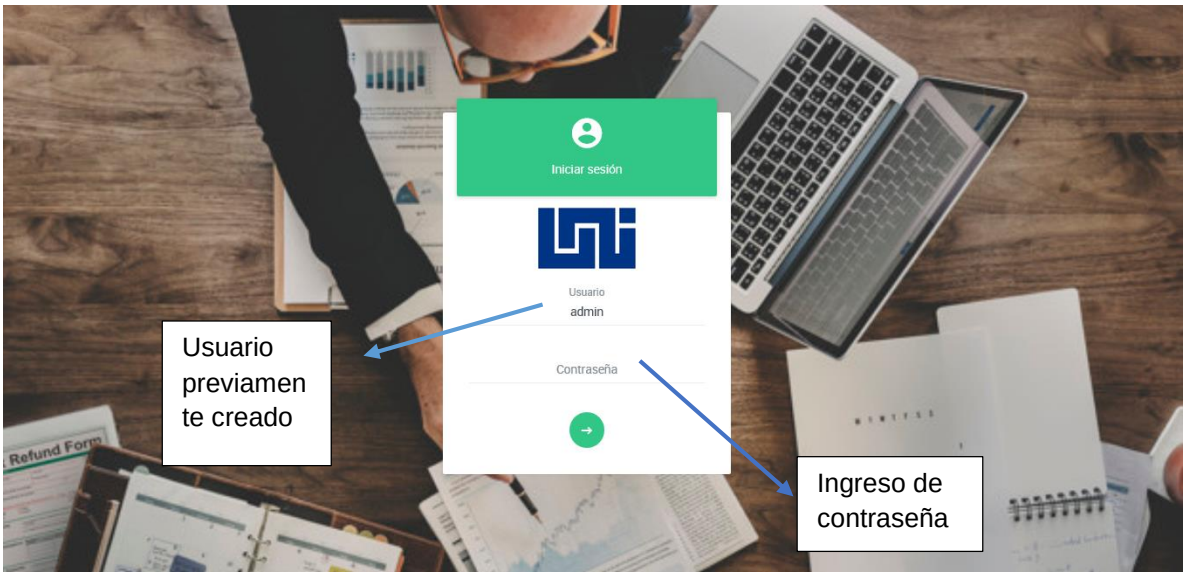


Figura No.1

Se ingresa el nombre del usuario y su respectiva contraseña, en este caso se accede con el usuario admin quien tiene todos los permisos de administración del sitio.

Una vez que el usuario y su contraseña son introducidas y no se produce ningún error en el acceso, la siguiente pantalla que aparecerá es la que se muestra a continuación.

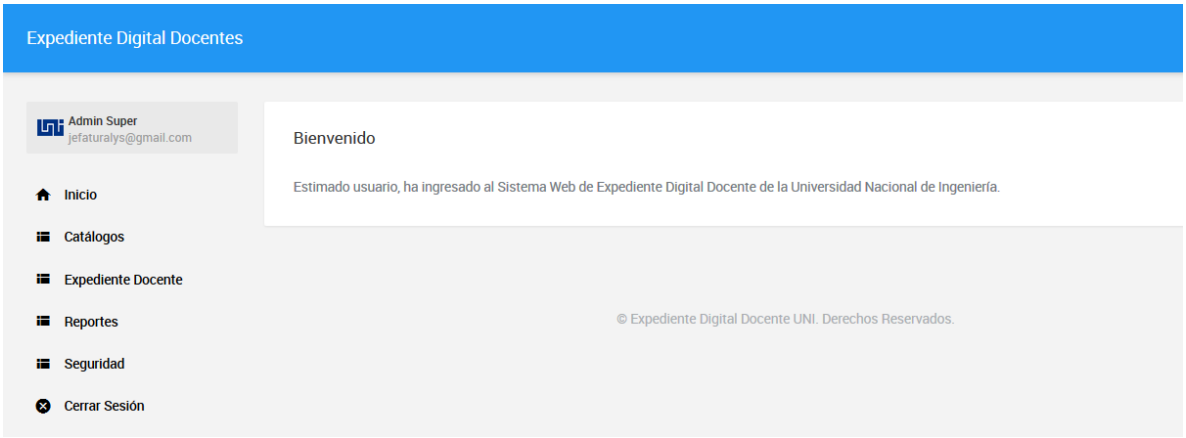


Figura No.2

Esta pantalla en caso del usuario admin, muestra las opciones de inicio, la cual permitirá regresar a la pantalla mostrada en dicha figura, la opción catalogo despliega las opciones del mismo, en ellas se pueden encontrar, por ejemplo: los recintos, facultades, carreras y departamentos docentes, a como se muestra a continuación.

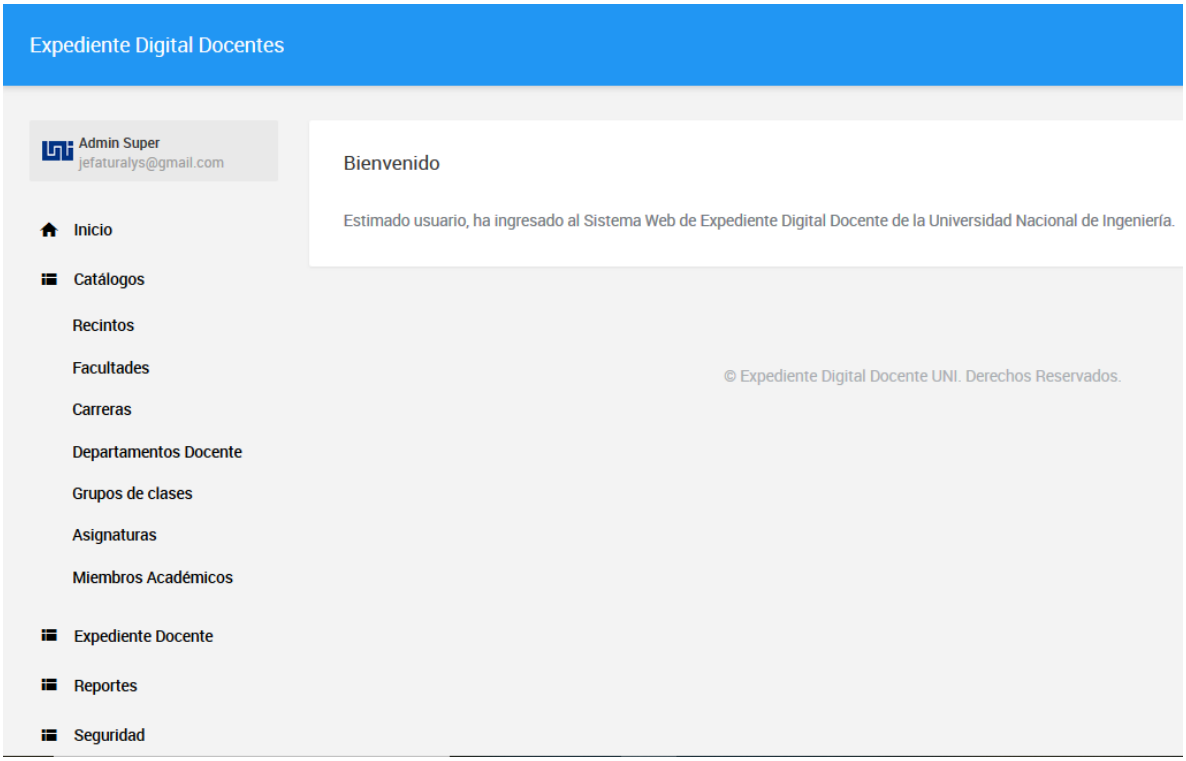


Figura No.3

Es importante mencionar que al hacer uso del diseño responsive, se pretende facilitar a los usuarios el acceso a la aplicación, por lo que la misma no está cargada de gráficos, a fin de hacer un uso adecuado de los recursos del dispositivo desde donde se acceda a la aplicación.

Si por el contrario en lugar de seleccionar la opción catálogo, se da clic sobre expediente de asignatura, se desplegarán las opciones que están dentro de ella, tal como se muestra en la pantalla que a continuación se tiene.

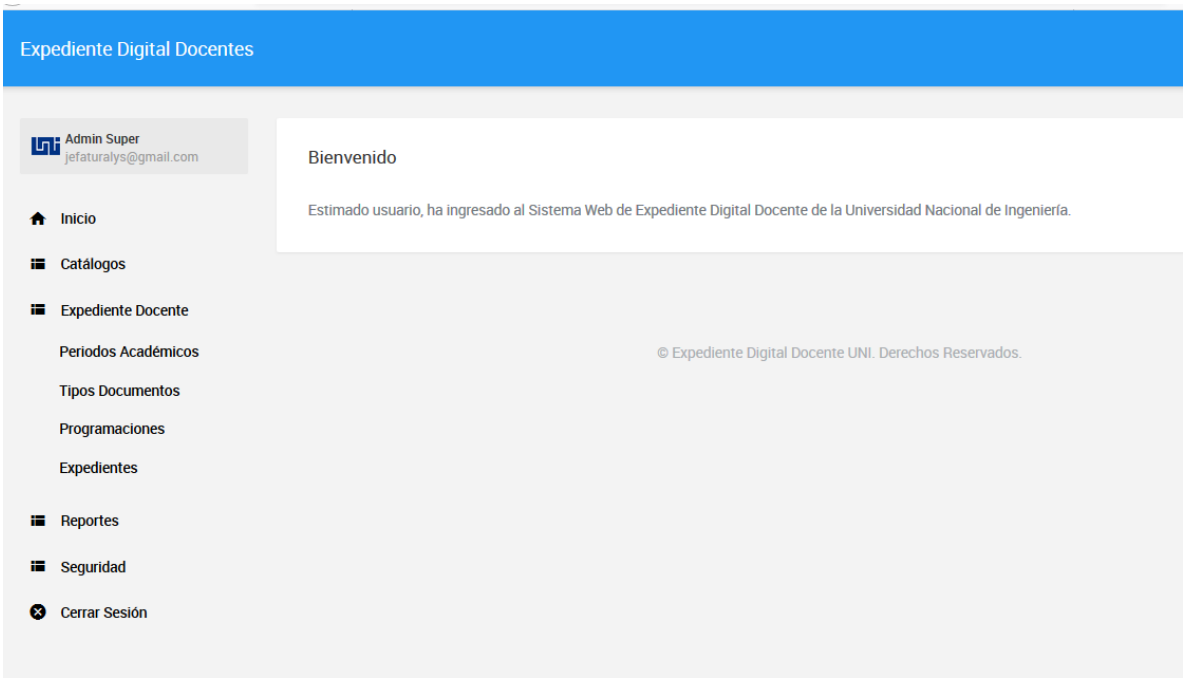


Figura No. 5

El expediente de asignatura, tiene un periodo académico que se debe seleccionar podría ser, por ejemplo: el semestre, o bien un curso de verano. También tiene la opción de tipo de documentos a subir, las programaciones que son los periodos que uno selecciona, así como el expediente en sí.

La opción reporte por su parte indica los informes que la aplicación puede presentar.

De igual manera existe la opción de seguridad la cual contiene los roles, permisos y usuarios, esta opción la utiliza el administrador

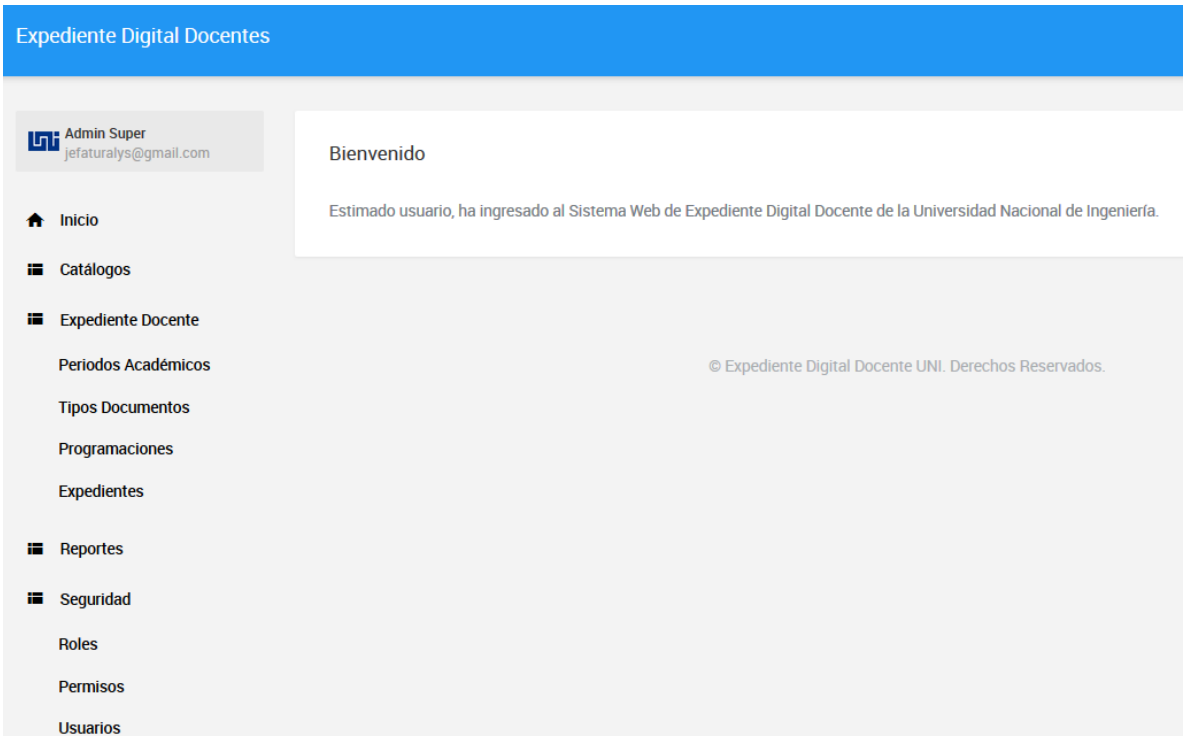


Figura No.6

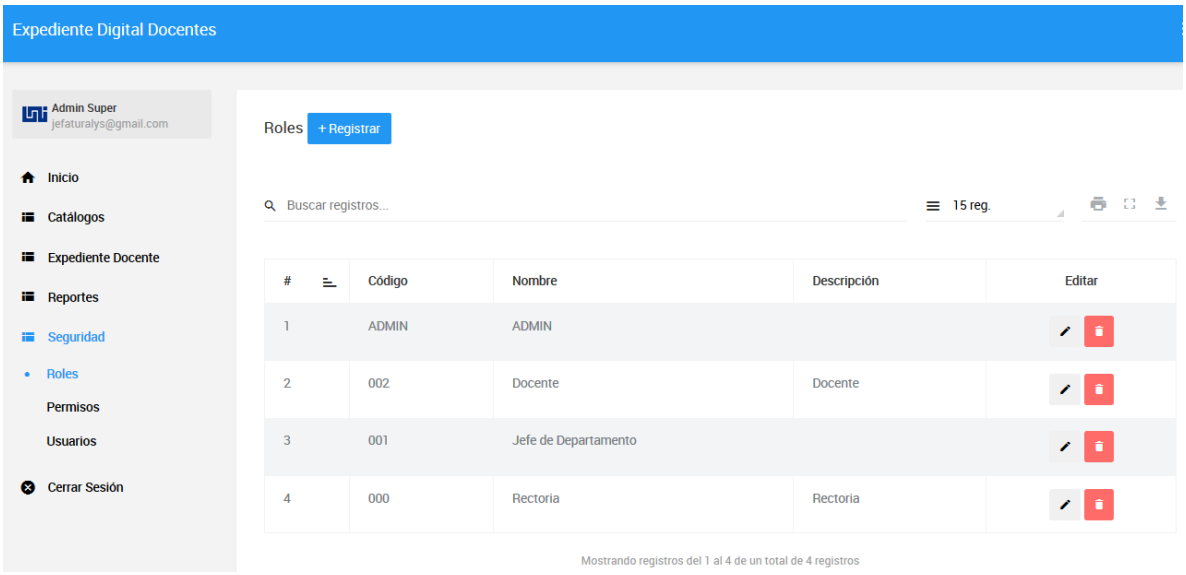


Figura No.7

Los roles están relacionados con las actividades que puede realizar un determinado usuario.

Los permisos por su parte están relacionados a lo que uno o varios usuarios le es permitido en una determinada área del sistema

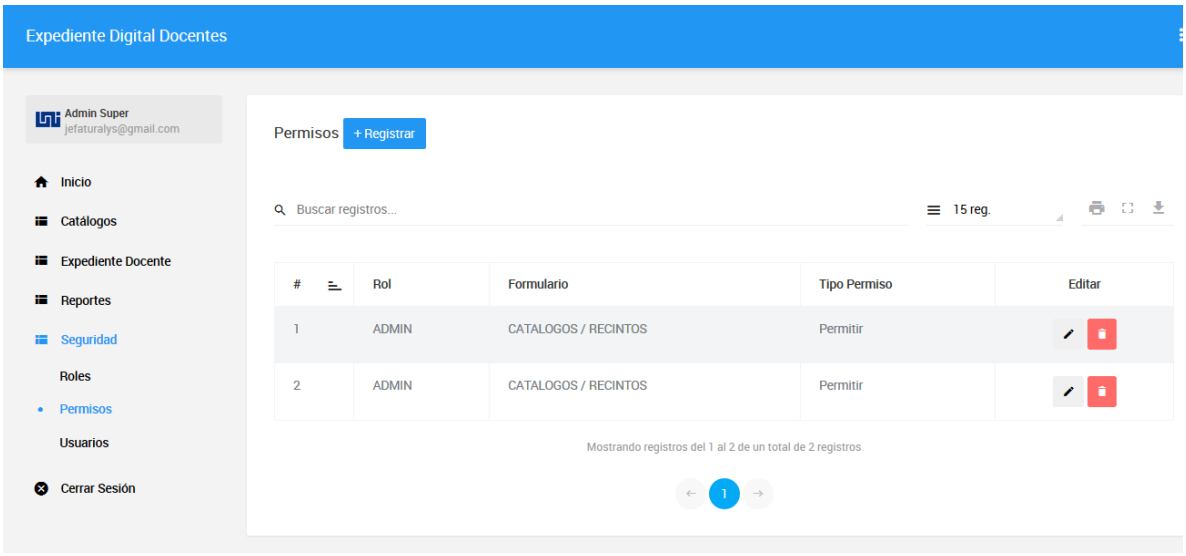


Figura No.8

La opción usuario permite la creación de un usuario por parte del administrador o bien por un usuario con permisos para realizarlo.

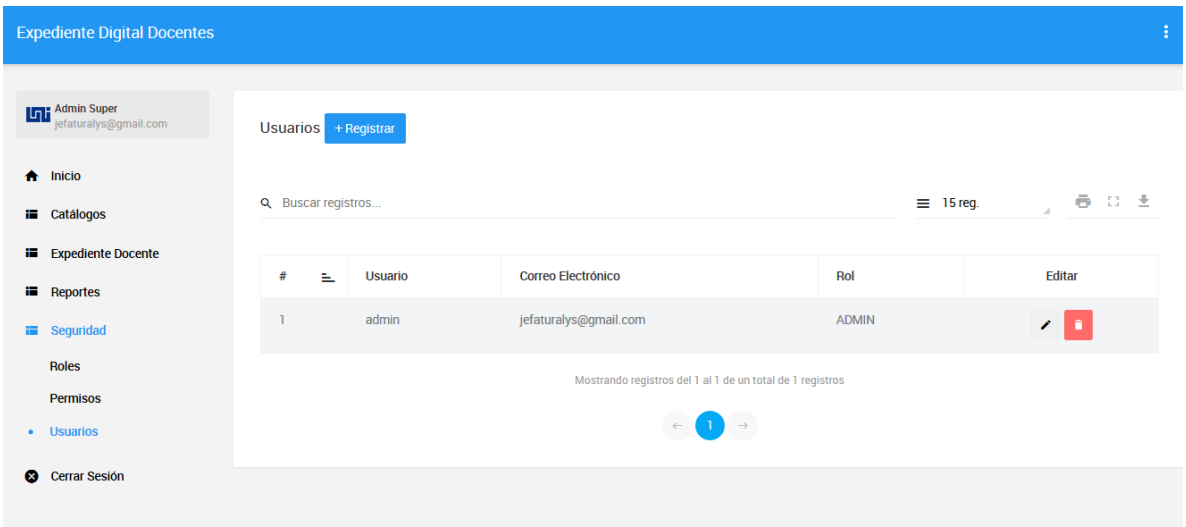


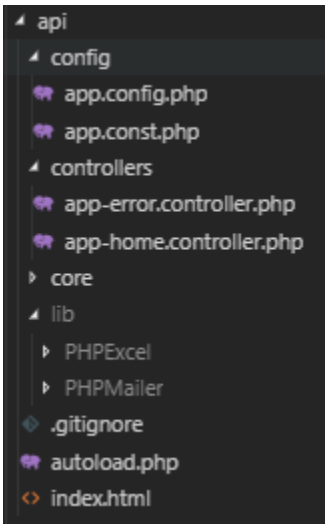
Figura No.9

La opción cerrar sesión permite que el usuario salga de forma correcta de la aplicación.

MANUAL DEL PROGRAMADOR

INTRODUCCIÓN

El propósito de este manual del programador, es dar a conocer al lector los códigos fuentes de la aplicación realizada. Para ello tratamos de la forma más concisa de explicar cada uno de los códigos, junto con la programación utilizada en el desarrollo del mismo, esto con el fin de que otros programadores puedan modificar, adecuar, en caso de ser necesaria la aplicación.



En el directorio **core** se encuentran las clases de los componentes del paquete **api : php**.

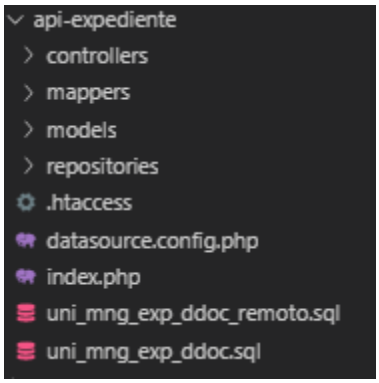
Directorio **lib** contiene librerías externas, en éste caso para la manipuliación de ficheros excel y el envío de correos.

autoload.php es el script que se importa desde api-rt-nc para el uso de todos los componentes basicos del servicio para su implementación conforme el diseño de tablas de la base de datos.

app.config y **app.const** contienen variables globales de configuración.

app-error.controlller y **app-home.controller** son controladores de uso en común para mostrar errores o url base respectivamente del servicio web.

Proyecto **api-expediente**



Contiene la implementación de los componentes de **api : php** por cada tabla del diseño de la base de datos, seguiendo la siguiente nomenclatura:

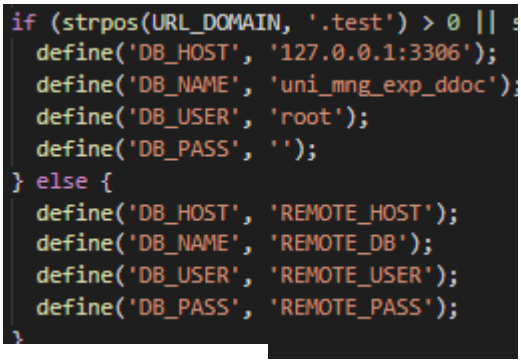
- **Archivos**: app-nombre-tabla.[model|mapper|repository|controller].php

- **Clases**: AppNombreTabla [Model|Mapper|Repository|Controller]

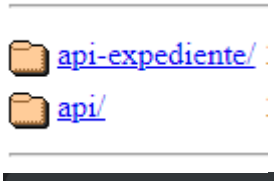
.htaccess: para la transformación dinámica de url amigables.
index.php: script inicial de la aplicación.
datasource.config.php: contiene la configuración de conexión a la base de datos.
uni_mng_exp_ddoc_remoto.sql: toda la definición de tablas de la base de datos, adaptada para su importación en el servidor de publicación.
uni_mng_exp_ddoc.sql: toda la definición de tablas de la base de datos, para uso local en el ambiente de desarrollo.

- **Configuración Inicial**

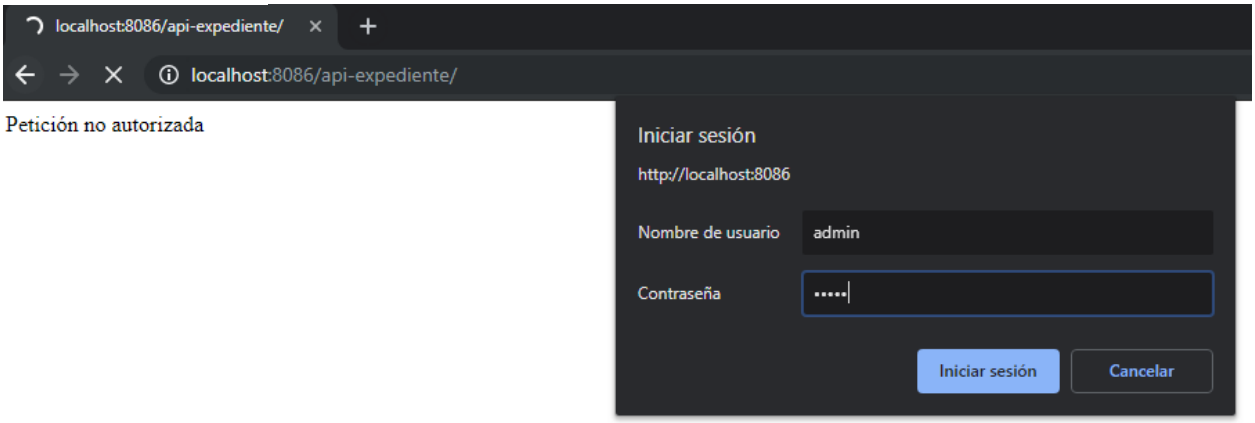
Lo primero que debe configurar es el archivo ubicado en **api-expediente/datasource.config.php**:



Corresponden a las variables de conexión a la base de datos tanto del servidor local, como del servidor remote (una vez se publique el servicio en el internet)
Una vez configuradas las variables, el servicio web queda listo para su funcionamiento.



Una vez cargados ambos directorios en el servidor de aplicaciones (en éste caso Apache HTTP/PHP), se debe arceder al **api-expediente**:



Una vez se accede en **api-expediente**, nos solicitará autenticación básica, en éste caso ingresaremos las credenciales (de lo contrario nos mostraría el mensaje de “Petición no autorizada”):

Usuario: **admin**
Contraseña: **admin**
En la url http://localhost:TU-PUERTO/api-expediente/ solamente nos mostrará una página en blanco, en éste caso ingresaremos al recurso http://localhost:TU-PUERTO/api-expediente/cat-recinto (conforme nomenclatura de cada controlador de las tablas de la base de datos) para consultar los recintos que estén registrados:
Obtenemos [] ya que en la tabla de cat_recintos no hay registros, una vez registrados algunos datos de pruebas, obtendríamos por ejemplo:

[{"id":"1","codigo":"RUSB","nombre":"RUSB","descripcion":"RUSB","estado":"1"}]

Tambien es necesario configurar las credenciales de un correo para el reseteo de contraseñas de usuario en el **index.php** la función **fnGenerateResetPasswordEmail**:

```php
function fnGenerateResetPasswordEmail($email, $passwordResetToken) {
  try {
    $mail = new PHPMailer(true);
    $mail->CharSet = "UTF-8";
    $mail->IsSMTP();
    $mail->Host = "smtp-mail.outlook.com";
    $mail->Port = 587;
    $mail->SMTPSecure = "tls";

    $mail->SMTPAuth = true;
    $mail->From = "TU_CORREO";
    $mail->Username = "TU_CORREO";
    $mail->Password = "TU_CLAVE";
```

- Detalle de métodos permitidos y estructura de cuerpo solicitudes (http request) y respuestas (http response).

Método	Descripción	Cuerpo Solicitud	Cuerpo Respuesta
GET	Obtener	No aplica	[{REGISTROS OBTENIDOS }]
POST	Registrar	{ data: [NUEVO REGISTRO] }	[{ID INSERTADO }]
PUT	Actualizar	{ data: [REGISTRO A ACTUALIZAR] }	[{ID ACTUALIZADO }]
DELETE	Eliminar	{ data: [REGISTRO A ELIMINAR] }	[{ID ELIMINADO }]

En el caso que se genere un error, el servicio web genera una estructura de respuesta con el detalle del error generado similar al siguiente:

{ "result": 1, "error": { "id": -2, "message": "No existe registro con id 9" } }	result 1: error generado, 0: ejecución exitosa; error.id: id del error (-1 general, -2 base de datos); error.message: detalle del error generado

- **Sitio Web Laravel 5.3.0**

Se efectuó la implementación del sitio web con Laravel para seguir los lineamiento de un Código bien estructurado y facilitar futuras integraciones, la estructura de directorios está

disponible en el sitio official del framework (https://laravel.com/docs/5.3/structure), para el caso de éste Proyecto los directorios y ficheros de importancia son los siguientes:

- **./ expediente-lrvl / .env**

Archivo de configuración del ambiente, en éste caso solamente fue necesario configurar los siguientes parámetros:

```
APP_ENV=local
APP_KEY=base64:VRfZwzpgUa5us+6fBZVK1kvPcfGWIPfgADXsWr9BQ6k=
APP_DEBUG=true
APP_LOG_LEVEL=debug
APP_URL=http://simedianet.com/expediente
```

No hubo necesidad de configurar la conexión a la base de datos, ya que se reutilizan las clases de repositorios de **api-expediente** para la administración de las tablas desde el sitio web.

- **./ expediente-lrvl / routes / web.php**

Sirve para configurar todas las rutas públicas que el usuario podrá acceder, inclusive indicando qué método http se debe permitir, por ejemplo:

```
Route::get('/usuarios', array('uses' => 'UsuariosController@mostrar'));

Route::post('/iniciar', array('uses' => 'IniciarController@procesar'));
```

- **./ expediente-lrvl / app / Http / Controllers / *.php**

Contiene todas las clases que controlarán los datos que se mostrarán y el procesamiento de los mismos para las operaciones CRUD de todos los formularios, de forma general tienen una método <mostrar> para la visualización de los datos y un método <procesar> para el tratamiento de las operaciones de inserción, actualización y eliminación.

- **./ expediente-lrvl / resources / views / [*.php, editores/*.php, default/*.php]**

Contiene toda la renderización de las vistas html de los formularios, para tal efecto:
> En la raíz de **/views/** se encuentran los scripts para la visualización de registros existentes (tablas de registros) y el formulario de inicio de sesión.
> En **editores/** la estructura de campos requeridos para la inserción y actualización de datos.
> **default/** plantillas de reutilización general como inclusión de estilos css/javascript, el menu superior, menú lateral izquierdo, y pie de página.

- **./ expediente-lrvl / public /**

Directorio de inicio de la aplicación: contiene todos los recursos de estilos css, imágenes, fuentes, y librerías javascript utilizadas en el sitio web, en el index.php se incluyó lo siguiente para la reutilización de los repositorios de base de datos del servicio web:

```php
define("DIR_BASE", "../../");
function gblIsSite() {
    return session()->get('site') == 1;
}
```

La constante DIR_**BASE** es para configurar el directorio raiz de los tres paquetes (./api/ , ./api-expediente/ y ./expediente-lrvl/).
La función **gblIsSite** es llamada desde **api-expediente** para detectar si se está reutilizando el servicio desde el sitio web, de tal forma, que los tres directorios (./api/ , ./api-expediente/ y ./expediente-lrvl/) deben de estar publicados en el mismo directorio raiz del dominio público y/o ambiente de desarrollo local.

API EXPEDIENTE – CONTROLLER

```php
<?php
class AppCatAsignaturaController extends AppController {
}
<?php
class AppCatCargoController extends AppController {
}
<?php
class AppCatCarreraController extends AppController {
}
<?php
class AppCatDepartamentoController extends AppController {
}
<?php
class AppCatFacultadController extends AppController {
}
<?php
class AppCatGrupoController extends AppController {
}
<?php
class AppCatMiembroController extends AppController {
}
<?php
class AppCatPersonaController extends AppController {
}
<?php
class AppCatRecintoCarreraController extends AppController {
```

```
        }
        <?php
        class AppCatRecintoController extends AppController {
        }
        <?php
        class AppPrtAsignaturaController extends AppController {
        }
        <?php
        class AppPrtDocumentoController extends AppController {
        }
        <?php
        class AppPrtExpdRevisionController extends AppController {
        }
        <?php
        class AppPrtExpedienteDocumentoController extends
AppController {
        }
        <?php
        class AppPrtExpedienteController extends AppController {
        }
        <?php
        class AppPrtPeriodoAcademicoController extends AppController
{
        }
        <?php
        class AppPrtProgramacionController extends AppController {
        }
        <?php
        class AppPrtTipoPeriodoController extends AppController {
        } <?php

        class AppSegFormularioOperacionController extends
AppController {
        }
        <?php
        class AppSegFormularioController extends AppController {
            /*public function get($id, $sid, $qry) {
              if ( ($id == null || $id < 1) && (!is_array($qry) ||
!isset($qry['modulo'])) )
                { throw new Exception('id modulo es requerido'); }
```

```
      return parent::get($id, $sid, $qry);
    }*/
}
<?php
class AppSegModuloController extends AppController {
}
<?php
class AppSegOperacionController extends AppController {
}
<?php
class AppSegRolPermisoController extends AppController {
}
<?php
class AppSegRolController extends AppController {
}
<?php
class AppSegUsuarioRolController extends AppController {
}
<?php
class AppSegUsuarioController extends AppController {
}
```

API EXPEDIENTE – MAPPERS

```
<?php
class AppCatAsignaturaMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppCatCargoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
```

```
class AppCatCarreraMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppCatDepartamentoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppCatFacultadMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppCatGrupoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppCatMiembroMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
```

```
class AppCatPersonaMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppCatRecintoCarreraMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php

class AppCatRecintoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppPrtAsignaturaMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppPrtDocumentoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
```

```php
<?php
class AppPrtExpdRevisionMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppPrtExpedienteDocumentoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppPrtExpedienteMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppPrtPeriodoAcademicoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppPrtProgramacionMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
```

```
<?php
class AppPrtTipoPeriodoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppSegFormularioOperacionMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppSegFormularioMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php

class AppSegModuloMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppSegOperacionMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
```

```
}
<?php
class AppSegRolPermisoMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppSegRolMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php
class AppSegUsuarioRolMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
}
<?php

class AppSegUsuarioMapper extends AppMapper {
  public function mapRow($rs, $rn) {
    return $rs;
  }
  public function mapRowRef(&$rs, $rn) {
  }
  }
```

API EXPEDIENTE – MODELS

```
<?php
class AppCatAsignaturaModel extends AppModel {
  public $id;
  public $departamento_id;
```

```
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php

class AppCatCarreraModel extends AppModel {
  public $id;
  public $facultad_id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppCatDepartamentoModel extends AppModel {
  public $id;
  public $recinto_id;
  public $facultad_id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppCatFacultadModel extends AppModel {
  public $id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppCatGrupoModel extends AppModel {
  public $id;
  public $recinto_id;
  public $carrera_id;
  public $codigo;
  public $nombre;
```

```
  public $descripcion;
  public $estado = 1;
}
<?php
class AppCatMiembroModel extends AppModel {
  public $id;
  public $facultad_id;
  public $departamento_id;
  public $persona_id;
  public $cargo_id;
  public $fecha_ini;
  public $fecha_fin;
  public $estado = 1;
}
<?php
class AppCatPersonaModel extends AppModel {
  public $id;
  public $cedula;
  public $nombres;
  public $apellidos;
  public $genero;
  public $estado = 1;
}
<?php
class AppCatRecintoCarreraModel extends AppModel {
  public $id;
  public $recinto_id;
  public $carrera_id;
  public $estado = 1;
}
<?php
class AppCatRecintoModel extends AppModel {
  public $id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppPrtAsignaturaModel extends AppModel {
```

```
  public $id;
  public $expediente_id;
  public $recinto_id;
  public $facultad_id;
  public $carrera_id;
  public $grupo_id;
  public $asignatura_id;
  public $estado = 1;
}
<?php
class AppPrtDocumentoModel extends AppModel {
  public $id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppPrtExpdRevisionModel extends AppModel {
  public $id;
  public $exp_documento_id;
  public $usuario_id;
  public $miembro_id;
  public $observaciones;
  public $fecha_reg;
  public $estado = 1;
}
<?php
class AppPrtExpedienteDocumentoModel extends AppModel {
  public $id;
  public $expediente_id;
  public $documento_id;
  public $prt_asignatura_id;
  public $archivo;
  public $fecha_reg;
  public $estado = 1;
}
<?php

class AppPrtExpedienteModel extends AppModel {
```

```
  public $id;
  public $programacion_id;
  public $usuario_id;
  public $miembro_id;
  public $codigo;
  public $descripcion;
  public $fecha_reg;
  public $fecha_ini;
  public $fecha_fin;
  public $estado = 1;
  function __construct() {
    $this->codigo = strtoupper(appGenerateRandomString(10));
    $this->fecha_reg = date('Y-m-d H:m:s');
  }
}
<?php
class AppPrtPeriodoAcademicoModel extends AppModel {
  public $id;
  public $sup_id;
  public $tipo_id;
  public $codigo;
  public $descripcion;
  public $fecha_ini;
  public $fecha_fin;
  public $estado = 1;
  function __construct() {
    $this->codigo = strtoupper(appGenerateRandomString(10));
  }
}
<?php
class AppPrtProgramacionModel extends AppModel {
  public $id;
  public $departamento_id;
  public $periodo_id;
  public $codigo;
  public $descripcion;
  public $fecha_ini;
  public $fecha_fin;
  public $estado = 1;
  function __construct() {
```

```
    $this->codigo = strtoupper(appGenerateRandomString(10));
  }
}
<?php

class AppPrtTipoPeriodoModel extends AppModel {
  public $id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php

class AppSegFormularioOperacionModel extends AppModel {
  public $id;
  public $formulario_id;
  public $operacion_id;
  public $estado = 1;
}
<?php
class AppSegFormularioModel extends AppModel {
  public $id;
  public $modulo_id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppSegModuloModel extends AppModel {
  public $id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppSegOperacionModel extends AppModel {
  public $id;
```

```
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppSegRolPermisoModel extends AppModel {
  public $id;
  public $rol_id;
  public $formulario_id;
  public $operacion_id;
  public $tipo_permiso;
  public $estado = 1;
}
<?php
class AppSegRolModel extends AppModel {
  public $id;
  public $codigo;
  public $nombre;
  public $descripcion;
  public $estado = 1;
}
<?php
class AppSegUsuarioRolModel extends AppModel {
  public $id;
  public $usuario_id;
  public $rol_id;
  public $estado = 1;
}
<?php

class AppSegUsuarioModel extends AppModel {
  public $id;
  public $persona_id;
  public $usuario;
  public $clave;
  public $clave_reset;
  public $correo;
  public $estado = 1;
}
```

```
<?php
```

API EXPEDIENTE – MAPPERS

```
class AppCatAsignaturaRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatAsignatura', 'cat_asignaturas');
  }

  public function select($id, $sid, $qry, $asArray = false) {
    $lid = ($id == null || $id < 1 ? null : $id);
    $params = array(':id' => $lid);
    $sql = "SELECT a.*, b.nombre as departamento
    FROM cat_asignaturas a
      INNER JOIN cat_departamentos b on b.id = a.departamento_id
    WHERE (:id IS NULL OR a.id = :id)";
    $query = $this->dbc->database->prepare($sql);
    $query->execute($params);
    $data = $query->fetchAll(PDO::FETCH_ASSOC);
    if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
    $rst = array();
    foreach ($data as $clave => $valor) {
      $lmdl = new $this->model_name();
      $lmdl->loadFromArray($valor);
      $this->mapper->mapRowRef($lmdl, 0);
      $lmdl->departamento = $valor['departamento'];
      $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
    }
    return $rst;
  }
}
<?php
```

```
class AppCatCargoRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatCargo', 'cat_cargos');
  }
}
<?php

class AppCatCarreraRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatCarrera', 'cat_carreras');
  }

  public function select($id, $sid, $qry, $asArray = false) {
    $lid = ($id == null || $id < 1 ? null : $id);
    $params = array(':id' => $lid);
    $sql = "SELECT a.id, a.facultad_id, a.codigo, a.nombre,
a.descripcion, a.estado, b.nombre as facultad
      , GROUP_CONCAT(d.nombre SEPARATOR ', ') as recintos
      , GROUP_CONCAT(d.id SEPARATOR ',') as recintos_id
    FROM cat_carreras a
      INNER JOIN cat_facultades b on b.id = a.facultad_id
      LEFT  JOIN cat_recinto_carreras c on c.carrera_id = a.id
      LEFT  JOIN cat_recintos d on d.id = c.recinto_id
    WHERE (:id IS NULL OR a.id = :id)
    GROUP BY a.id, a.facultad_id, a.codigo, a.nombre, a.descripcion,
a.estado, b.nombre ";
    $query = $this->dbc->database->prepare($sql);
    $query->execute($params);
    $data = $query->fetchAll(PDO::FETCH_ASSOC);
    if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
    $rst = array();
    foreach ($data as $clave => $valor) {
      $lmdl = new $this->model_name();
      $lmdl->loadFromArray($valor);
      $this->mapper->mapRowRef($lmdl, 0);
      $lmdl->facultad = $valor['facultad'];
      $lmdl->recintos = $valor['recintos'];
      $lmdl->recintos_id = $valor['recintos_id'];
      $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
```

```
    }
    return $rst;
  }
}
<?php

class AppCatDepartamentoRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatDepartamento', 'cat_departamentos');
  }

  public function select($id, $sid, $qry, $asArray = false) {
    $lid = ($id == null || $id < 1 ? null : $id);
    $params = array(':id' => $lid);
    $sql = "SELECT a.*, b.nombre as recinto, c.nombre as facultad
    FROM cat_departamentos a
      INNER JOIN cat_recintos b on b.id = a.recinto_id
      INNER JOIN cat_facultades c on c.id = a.facultad_id
    WHERE (:id IS NULL OR a.id = :id)";
    $query = $this->dbc->database->prepare($sql);
    $query->execute($params);
    $data = $query->fetchAll(PDO::FETCH_ASSOC);
    if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
    $rst = array();
    foreach ($data as $clave => $valor) {
      $lmdl = new $this->model_name();
      $lmdl->loadFromArray($valor);
      $this->mapper->mapRowRef($lmdl, 0);
      $lmdl->recinto = $valor['recinto'];
      $lmdl->facultad = $valor['facultad'];
      $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
    }
    return $rst;
  }
}
<?php

class AppCatFacultadRepository extends AppRepository {
  public function __construct($db) {
```

```
    parent::__construct($db, 'CatFacultad', 'cat_facultades');
  }
}
<?php

class AppCatGrupoRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatGrupo', 'cat_grupos');
  }

  public function select($id, $sid, $qry, $asArray = false) {
    $lid = ($id == null || $id < 1 ? null : $id);
    $params = array(':id' => $lid);
    $sql = "SELECT a.*, b.nombre as recinto, c.nombre as carrera
    FROM cat_grupos a
      INNER JOIN cat_recintos b on b.id = a.recinto_id
      INNER JOIN cat_carreras c on c.id = a.carrera_id
    WHERE (:id IS NULL OR a.id = :id)";
    $query = $this->dbc->database->prepare($sql);
    $query->execute($params);
    $data = $query->fetchAll(PDO::FETCH_ASSOC);
    if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
    $rst = array();
    foreach ($data as $clave => $valor) {
      $lmdl = new $this->model_name();
      $lmdl->loadFromArray($valor);
      $this->mapper->mapRowRef($lmdl, 0);
      $lmdl->recinto = $valor['recinto'];
      $lmdl->carrera = $valor['carrera'];
      $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
    }
    return $rst;
  }
}
<?php

class AppCatMiembroRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatMiembro', 'cat_miembros');
```

```
  }

  public function select($id, $sid, $qry, $asArray = false) {
    $lid = ($id == null || $id < 1 ? null : $id);
    $params = array(':id' => $lid);
    $xtrSql = "";
    if ($qry != null && isset($qry['persona'])) {
      $xtrSql = " AND a.persona_id = :persona_id ";
      $params[':persona_id'] = $qry['persona'];
    }
    $sql = "SELECT a.*, b.nombre as facultad
      , c.nombre as departamento
      , CONCAT(d.nombres, ' ', d.apellidos) as persona
      , d.cedula, e.nombre as cargo, e.codigo as cargo_codigo
      , CAST(a.fecha_ini as DATE) as fecha_ini
      , CAST(a.fecha_fin as DATE) as fecha_fin
    FROM cat_miembros a
      LEFT JOIN cat_facultades b on b.id = a.facultad_id
      LEFT JOIN cat_departamentos c on c.id = a.departamento_id
      LEFT JOIN cat_personas d on d.id = a.persona_id
      LEFT JOIN cat_cargos e on e.id = a.cargo_id
    WHERE (:id IS NULL OR a.id = :id) $xtrSql ";
    $query = $this->dbc->database->prepare($sql);
    $query->execute($params);
    $data = $query->fetchAll(PDO::FETCH_ASSOC);
    if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
    $rst = array();
    foreach ($data as $clave => $valor) {
      $lmdl = new $this->model_name();
      $lmdl->loadFromArray($valor);
      $this->mapper->mapRowRef($lmdl, 0);
      $lmdl->facultad = $valor['facultad'];
      $lmdl->departamento = $valor['departamento'];
      $lmdl->persona = $valor['persona'];
      $lmdl->cedula = $valor['cedula'];
      $lmdl->cargo = $valor['cargo'];
      $lmdl->cargo_codigo = $valor['cargo_codigo'];
      $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
    }
```

```
    return $rst;
  }
}
<?php

class AppCatPersonaRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatPersona', 'cat_personas');
  }
}
<?php

class AppCatRecintoCarreraRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'CatRecintoCarrera',
'cat_recinto_carreras');
  }

  public function updateByCarrera($carrera_id, $recintos_id) {
    $sql = "DELETE FROM cat_recinto_carreras WHERE carrera_id =
:carrera_id ";
    $query = $this->dbc->database->prepare($sql);
    $query->execute(array(':carrera_id' => $carrera_id));
    $ndt = array();
    foreach ($recintos_id as $rec_id) {
      $ndt[] = array(
        'recinto_id' => $rec_id,
        'carrera_id' => $carrera_id
      );
    }
    $this->insert($ndt);
  }}
        <?php

        class AppCatRecintoRepository extends AppRepository {
          public function __construct($db) {
            parent::__construct($db, 'CatRecinto', 'cat_recintos');
          }
        }
        <?php
```

```
class AppPrtAsignaturaRepository extends AppRepository {
  public function __construct($db) {
    parent::__construct($db, 'PrtAsignatura', 'prt_asignaturas');
  }

  public function select($id, $sid, $qry, $asArray = false) {
    $lid = ($id == null || $id < 1 ? null : $id);
    $params = array(':id' => $lid);
    $sql = "SELECT a.*
    FROM prt_asignaturas a
    WHERE (:id IS NULL OR a.id = :id)";
    if ($qry != null && isset($qry['expediente'])) {
      $sql = "SELECT a.*
      FROM prt_asignaturas a
      WHERE (:id IS NULL OR a.id = :id) AND a.expediente_id =
:expediente_id";
      $params[':expediente_id'] = $qry['expediente'];
    }
    $query = $this->dbc->database->prepare($sql);
    $query->execute($params);
    $data = $query->fetchAll(PDO::FETCH_ASSOC);
    if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
    $rst = array();
    foreach ($data as $clave => $valor) {
      $lmdl = new $this->model_name();
      $lmdl->loadFromArray($valor);
      $this->mapper->mapRowRef($lmdl, 0);
      $lmdl->periodo = $valor['periodo'];
      $lmdl->departamento = $valor['departamento'];
      $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
    }
    return $rst;
  }
}
<?php

class AppPrtDocumentoRepository extends AppRepository {
  public function __construct($db) {
```

```
        parent::__construct($db, 'PrtDocumento', 'prt_documentos');
      }}
      <?php

    class AppPrtExpdRevisionRepository extends AppRepository {
      public function __construct($db) {
        parent::__construct($db, 'PrtExpdRevision',
'prt_expd_revisiones');
      }

      public function select($id, $sid, $qry, $asArray = false) {
        if (($id == null || $id == 0) && (!$qry ||
!isset($qry['documento']))) { return array(); }
        $lid = ($id == null || $id < 1 ? null : $id);
        $params = array(':id' => $lid, );
        $sql = "SELECT a.*
        FROM prt_expd_revisiones a
        WHERE (:id IS NULL OR a.id = :id)";
        if (isset($qry['documento'])) {
          $sql = "SELECT a.*
          FROM prt_expd_revisiones a
          WHERE (:id IS NULL OR a.id = :id) AND
a.exp_documento_id = :documento_id
          ORDER BY id DESC;";
          $params[':documento_id'] = $qry['documento'];
        }
        $query = $this->dbc->database->prepare($sql);
        $query->execute($params);
        $data = $query->fetchAll(PDO::FETCH_ASSOC);
        if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
        $rst = array();
        foreach ($data as $clave => $valor) {
          $lmdl = new $this->model_name();
          $lmdl->loadFromArray($valor);
          $this->mapper->mapRowRef($lmdl, 0);
          $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
        }
        return $rst;
      }
```

```
        }
        <?php

        class AppPrtExpedienteDocumentoRepository extends
AppRepository {
          public function __construct($db) {
            parent::__construct($db, 'PrtExpedienteDocumento',
'prt_expediente_documentos');
          }

          public function select($id, $sid, $qry, $asArray = false) {
            if (($id == null || $id == 0) && (!$qry ||
!isset($qry['expediente']))) { return array(); }
            $lid = ($id == null || $id < 1 ? null : $id);
            $params = array(':id' => $lid, );
            $sql = "SELECT a.*, b.nombre as documento
              , (SELECT COUNT(1) FROM prt_expd_revisiones r WHERE
r.exp_documento_id = a.id) as revisiones
            FROM prt_expediente_documentos a
              INNER JOIN prt_documentos b ON b.id = a.documento_id
            WHERE (:id IS NULL OR a.id = :id)";
            if (isset($qry['expediente'])) {
              $sql = "SELECT a.*, b.nombre as documento
                , (SELECT COUNT(1) FROM prt_expd_revisiones r
WHERE r.exp_documento_id = a.id) as revisiones
              FROM prt_expediente_documentos a
                INNER JOIN prt_documentos b ON b.id = a.documento_id
              WHERE (:id IS NULL OR a.id = :id) AND a.expediente_id =
:expediente_id";
              $params[':expediente_id'] = $qry['expediente'];
            }
            $query = $this->dbc->database->prepare($sql);
            $query->execute($params);
            $data = $query->fetchAll(PDO::FETCH_ASSOC);
            if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
            $rst = array();
            foreach ($data as $clave => $valor) {
              $lmdl = new $this->model_name();
              $lmdl->loadFromArray($valor);
```

```
            $this->mapper->mapRowRef($lmdl, 0);
            $lmdl->documento = $valor['documento'];
            $lmdl->revisiones = $valor['revisiones'];
            $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
          }
          return $rst;
        }}
      <?php

      class AppPrtExpedienteRepository extends AppRepository {
        public function __construct($db) {
          parent::__construct($db, 'PrtExpediente', 'prt_expedientes');
        }

        public function select($id, $sid, $qry, $asArray = false) {
          $lid = ($id == null || $id < 1 ? null : $id);
          $params = array(':id' => $lid);
          $sql = "SELECT a.*
          , CAST(a.fecha_ini as DATE) as fecha_ini
          FROM prt_expedientes a
          WHERE (:id IS NULL OR a.id = :id)";
          $query = $this->dbc->database->prepare($sql);
          $query->execute($params);
          $data = $query->fetchAll(PDO::FETCH_ASSOC);
          if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
          $rst = array();
          foreach ($data as $clave => $valor) {
            $lmdl = new $this->model_name();
            $lmdl->loadFromArray($valor);
            $this->mapper->mapRowRef($lmdl, 0);
            //$lmdl->periodo = $valor['periodo'];
            $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
          }
          return $rst;
        }
      }
      <?php
```

```
        class AppPrtPeriodoAcademicoRepository extends
AppRepository {
          public function __construct($db) {
            parent::__construct($db, 'PrtPeriodoAcademico',
'prt_periodos_academicos');
          }

          public function select($id, $sid, $qry, $asArray = false) {
            $lid = ($id == null || $id < 1 ? null : $id);
            $params = array(':id' => $lid);
            $sql = "SELECT a.*, b.nombre as tipo
              , CAST(a.fecha_ini as DATE) as fecha_ini
              , CAST(a.fecha_fin as DATE) as fecha_fin
            FROM prt_periodos_academicos a
              INNER JOIN prt_tipos_periodos b on b.id = a.tipo_id
            WHERE (:id IS NULL OR a.id = :id)";
            $query = $this->dbc->database->prepare($sql);
            $query->execute($params);
            $data = $query->fetchAll(PDO::FETCH_ASSOC);
            if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
            $rst = array();
            foreach ($data as $clave => $valor) {
              $lmdl = new $this->model_name();
              $lmdl->loadFromArray($valor);
              $this->mapper->mapRowRef($lmdl, 0);
              $lmdl->tipo = $valor['tipo'];
              $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
            }
            return $rst;
          }}
        <?php

        class AppPrtProgramacionRepository extends AppRepository {
          public function __construct($db) {
            parent::__construct($db, 'PrtProgramacion',
'prt_programaciones');
          }

          public function select($id, $sid, $qry, $asArray = false) {
```

```
            $lid = ($id == null || $id < 1 ? null : $id);
            $params = array(':id' => $lid);
            $sql = "SELECT a.*, b.descripcion as periodo
              , c.nombre as departamento
              , CAST(a.fecha_ini as DATE) as fecha_ini
              , CAST(a.fecha_fin as DATE) as fecha_fin
            FROM prt_programaciones a
              LEFT JOIN prt_periodos_academicos b on b.id =
a.periodo_id
              LEFT JOIN cat_departamentos c on c.id = a.departamento_id
            WHERE (:id IS NULL OR a.id = :id)";
            if ($qry != null && isset($qry['usuario'])) {
              $sql = "SELECT a.*, b.descripcion as periodo
                , c.nombre as departamento
                , CAST(a.fecha_ini as DATE) as fecha_ini
                , CAST(a.fecha_fin as DATE) as fecha_fin
              FROM prt_programaciones a
                INNER JOIN cat_miembros mm on mm.departamento_id =
a.departamento_id
                INNER JOIN cat_personas pp on pp.id = mm.persona_id
                INNER JOIN seg_usuarios uu on uu.persona_id = pp.id
                LEFT JOIN prt_periodos_academicos b on b.id =
a.periodo_id
                LEFT JOIN cat_departamentos c on c.id =
a.departamento_id
              WHERE (:id IS NULL OR a.id = :id) AND uu.id = :usuario_id
";
              $params[':usuario_id'] = $qry['usuario'];
            }
            $query = $this->dbc->database->prepare($sql);
            $query->execute($params);
            $data = $query->fetchAll(PDO::FETCH_ASSOC);
            if ($lid !== null && count($data) < 1) { throw new Exception('No
existe registro con id ' . $lid); }
            $rst = array();
            foreach ($data as $clave => $valor) {
              $lmdl = new $this->model_name();
              $lmdl->loadFromArray($valor);
              $this->mapper->mapRowRef($lmdl, 0);
              $lmdl->periodo = $valor['periodo'];
```

```
            $lmdl->departamento = $valor['departamento'];
            $rst[$clave] = $asArray ? (array)$lmdl : $lmdl;
          }
          return $rst;
        }
      } <?php

      class AppPrtTipoPeriodoRepository extends AppRepository {
        public function __construct($db) {
          parent::__construct($db, 'PrtTipoPeriodo',
'prt_tipos_periodos');
        }}
```

DATA SOURCE

```
      <?php

      /** * Tipo de basse de datos */
      define('DB_TYPE', 'mysql');
      /** * Conjunto de caracteres */
      define('DB_CHARSET', 'utf8');
      /**
       * DB_HOST: servidor de la base de datos
       * DB_NAME: nombre del esquema
       * DB_USER: usuario de conexión
       * DB_PASS: contraseña del usuario
       */
      if (strpos(URL_DOMAIN, '.test') > 0 || strpos(URL_DOMAIN,
'localhost') >= 0) {
        define('DB_HOST', 'localhost');
        define('DB_NAME', 'uni_mng_exp_ddoc');
        define('DB_USER', 'simedian');
        define('DB_PASS', 'Psimedia1!!%');
      } else {
        define('DB_HOST', 'REMOTE_HOST');
        define('DB_NAME', 'REMOTE_DB');
        define('DB_USER', 'REMOTE_USER');
        define('DB_PASS', 'REMOTE_PASS');
      }
```